

042390.P10451

PATENT

UNITED STATES PATENT APPLICATION
FOR

ENHANCING ENTROPY IN PSEUDO-RANDOM NUMBER GENERATORS USING REMOTE
SOURCES

INVENTORS:

MATTHEW WOOD
a citizen of the United States,
residing at 7570 NE Chesapeake St.
Hillsboro, OR 97124

GARY GRAUNKE
a citizen of the United States,
residing at 362 NE Hillwood Drive
Hillsboro, OR 97124

PREPARED BY:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1026
(303) 740-1980

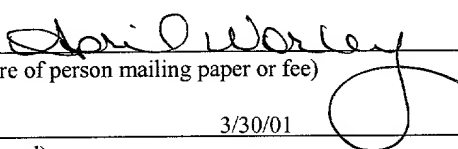
EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number: EL845313323US

Date of Deposit: March 30, 2001

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service
"Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has
been addressed to the Commissioner of Patents and Trademarks, Washington, D. C. 20231

April M. Worley
(Typed or printed name of person mailing paper or fee)


(Signature of person mailing paper or fee)

3/30/01
(Date signed)

ENHANCING ENTROPY IN PSEUDO-RANDOM NUMBER GENERATORS USING REMOTE SOURCES

COPYRIGHT NOTICE

[0001] Contained herein is material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction of the patent disclosure by any person as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all rights to the copyright whatsoever.

FIELD OF THE INVENTION

[0002] This invention relates to enhancing entropy, in general, and more specifically to entropy amplification in pseudo-random numbers using remote sources.

BACKGROUND OF THE INVENTION

[0003] Securing data through encryption/decryption methods, especially, when transmitting it over insecure channels, from cryptographic attacks is widely known. Traditionally, a method of symmetric encryption was used to secure the information between two users. The method of symmetric encryption required creating a single secret key known only to the two users. However, the secrecy was only guaranteed to the extent the two users kept the key secret. Additionally, the method of prior exchange of the key made the system even more cumbersome. To make the system more secure and reliable, the public-key system was introduced.

[0004] In a public-key system, also known as the asymmetric or two-key system, each user's key has a public and private component. The public component generates public encryption, while the private component generates private decryption of the encrypted text. This makes the system much more secure, because it is difficult to break an encryption, unless the corresponding private key is also known.

[0005] A typical public-key system uses a pseudo-random number generator (PRNG) to generate random numbers through a deterministic process. Consequently, the security of such system is dependent upon having a strong pseudo-random number generation (PRNG) algorithm. A PRNG uses a random internal state and a process called stirring to produce a stream of bits that satisfy various statistical tests of cryptographic randomness. The internal state is initialized with a random value called a seed. The seed must have a high level of entropy to ensure that the stream of bits are sufficiently hard to guess. Existing methods of gathering entropy use information gathered from a local system to seed the PRNG. If the seed gathered from the local system does not have sufficient entropy, an attacker can guess the output of the PRNG with relative ease, and break the system. This is especially true in constrained environments such as the Java Virtual Machine.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] The appended claims set forth the features of the invention with particularity. The invention, together with its advantages, may be best understood from the following detailed description taken in conjunction with the accompanying drawings of which:

[0007] Figure 1 is a block diagram of a typical computer system upon which one embodiment of the present invention may be implemented;

[0008] Figure 2 is a block diagram illustrating an exemplary network upon which the present invention may be implemented;

[0009] Figure 3 is a block diagram illustrating symmetric-key and public-key encryptions;

[0010] Figure 4 is a block diagram illustrating logic for using a set of redundant entropy servers, according to one embodiment of the present invention;

[0011] Figure 5 is a flow diagram illustrating the process for using a set of redundant entropy servers, according to one embodiment of the present invention;

[0012] Figure 6 is a block diagram illustrating logic for implementing a secure entropy collection protocol, according to one embodiment of the present invention;

[0013] Figure 7 is a flow diagram illustrating the process for implementing a secure entropy collection protocol, according to one embodiment of the present invention.

DETAILED DESCRIPTION

[0014] A method and apparatus are described for enhancing entropy in a pseudo-random number generator using a remote source. Broadly stated, embodiments of the present invention allows the stirring of a pseudo-random number generator using both the local seeding information and, for additional security, remote seeding information generated by remote entropy servers.

[0015] According to one embodiment, one or more remote entropy servers generate seeding information, which is securely gathered along with the local seeding information. An attacker can easily break into a system using only the local seeding information, and predict the state of a PRNG. The use of the remote seeding information adds to the randomness of the PRNG making a system much more secure from cryptographic attacks. Protecting systems from cryptographic attacks by enhancing entropy using remote sources can secure all types of transactions, such as emailing, banking transactions, and communication between applications.

[0016] In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some of these specific details. In other instances, well-known structures and devices are shown in block diagram form.

[0017] The present invention includes various steps, which will be described below. The steps of the present invention may be performed by hardware components or

may be embodied in machine-executable instructions, which may be used to cause a general-purpose or special-purpose processor or logic circuits programmed with the instructions to perform the steps. Alternatively, the steps may be performed by a combination of hardware and software.

[0018] The present invention may be provided as a computer program product, which may include a machine-readable medium having stored thereon instructions which may be used to program a computer (or other electronic devices) to perform a process according to the present invention. The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, CD-ROMs, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, flash memory, or other type of media / machine-readable medium suitable for storing electronic instructions. Moreover, the present invention may also be downloaded as a computer program product, wherein the program may be transferred from a remote computer to a requesting computer by way of data signals embodied in a carrier wave or other propagation medium via a communication link (e.g., a modem or network connection).

[0019] **Figure 1** is a block diagram of a typical computer system upon which one embodiment of the present invention may be implemented. Computer system 100 comprises a bus or other communication means 101 for communicating information, and a processing means such as processor 102 coupled with bus 101 for processing information. Computer system 100 further comprises a random access memory (RAM) or other dynamic storage device 104 (referred to as main memory), coupled to bus 101 for storing

information and instructions to be executed by processor 102. Main memory 104 also may be used for storing temporary variables or other intermediate information during execution of instructions by processor 102. Computer system 100 also comprises a read only memory (ROM) and/or other static storage device 106 coupled to bus 101 for storing static information and instructions for processor 102.

[0020] A data storage device 107 such as a magnetic disk or optical disc and its corresponding drive may also be coupled to computer system 100 for storing information and instructions. Computer system 100 can also be coupled via bus 101 to a display device 121, such as a cathode ray tube (CRT) or Liquid Crystal Display (LCD), for displaying information to an end user. Typically, an alphanumeric input device 122, including alphanumeric and other keys, may be coupled to bus 101 for communicating information and/or command selections to processor 102. Another type of user input device is cursor control 123, such as a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 102 and for controlling cursor movement on display 121.

[0021] A communication device 125 is also coupled to bus 101. The communication device 125 may include a modem, a network interface card, or other well-known interface devices, such as those used for coupling to Ethernet, token ring, or other types of physical attachment for purposes of providing a communication link to support a local or wide area network, for example. In this manner, the computer system 100 may be coupled to a number of clients and/or servers via a conventional network

infrastructure, such as a company's Intranet and/or the Internet, for example.

[0022] It is appreciated that a lesser or more equipped computer system than the example described above may be desirable for certain implementations. Therefore, the configuration of computer system 100 will vary from implementation to implementation depending upon numerous factors, such as price constraints, performance requirements, technological improvements, and/or other circumstances.

[0023] It should be noted that, while the steps described herein may be performed under the control of a programmed processor, such as processor 102, in alternative embodiments, the steps may be fully or partially implemented by any programmable or hard-coded logic, such as Field Programmable Gate Arrays (FPGAs), TTL logic, or Application Specific Integrated Circuits (ASICs), for example. Additionally, the method of the present invention may be performed by any combination of programmed general-purpose computer components and/or custom hardware components. Therefore, nothing disclosed herein should be construed as limiting the present invention to a particular embodiment wherein the recited steps are performed by a specific combination of hardware components.

[0024] **Figure 2** is a block diagram illustrating an exemplary network upon which the present invention may be implemented. In this example, an Ethernet network 210 is shown. Such a network may utilize Transmission Control Protocol/Internet Protocol (TCP/IP). Of course, many other types of networks and protocols are available and are commonly used. However, for illustrative purposes, Ethernet and TCP/IP will be

referred.

[0025] Connected to this network 210 is a local system 220. In addition to the local system 220, one or more remote independent systems 230 and 240 are connected to the network 210. As illustrated, the remote independent systems 230 and 240 include entropy servers 230 and 240. The number and arrangement of this equipment may vary depending on the application.

[0026] **Figures 3A and 3B** are block diagrams illustrating symmetric-key and public-key encryptions. As illustrated, the original data 305 is encrypted 315 using the symmetric-key 310. The same symmetric-key 310 is used to decrypt the data into its original form 320. The symmetric-key process 300 is extremely time- and processor-efficient, because only native processor instructions such as addition, bitwise logical-OR, bitwise logical-AND, and bitwise logical-exclusive-OR based on the key are used to encrypt and decrypt the text. However, the system is secured only to the extent that the two parties can keep the key secret.

[0027] In contrast, as illustrated by figure 3B, Public-key encryption 350 uses a public key 360, and a private key 370 to obtain the encrypted data 365 and decrypted data 375, respectively. In public-key encryption, the text 355 is encrypted 365, with the receiving party's public key 360. Upon reception, the receiver may decrypt 375 the encrypted text 365 using the corresponding private key 370. Since only the private key 370 is kept secret, while the public key 360 is openly distributed, the need for both parties to share a secret is eliminated.

[0028] Data is most often exchanged between parties encrypted with a symmetric key, and the symmetric key is encrypted with the public key of the receiving party and sent with the encrypted data. Thus, it has the performance benefits of symmetric encryption, with the advantages of public-key encryption. Encrypting a symmetric key with the recipient's public key is called a key exchange. The entire process of encrypting data with a symmetric key, encrypting the symmetric key, and sending encrypted data and encrypted symmetric key to the recipient is often referred to as "encrypting with the recipient's public key." It will be used this way for the rest of the description.

[0029] **Figure 4** is a block diagram illustrating logic for enhancing entropy using a set of redundant entropy servers (see figure 2), according to one embodiment of the present invention. As illustrated, a local system 405 comprises a pseudo-random number generator (PRNG) 415, and at least a system of gathering local seeding information 410, and the stirring process 425. However, the local system 405, according to one embodiment of the present invention, also gathers remote seeding information 420 generated by one or more remote entropy servers 430 and 445. The remote entropy servers 430 and 445 comprise a random state machine 435 and 450, and generate seeding information 440 and 455 to later stir the PRNG 425.

[0030] Generally, a PRNG uses a random internal state and the stirring process to produce a stream of bits that satisfy various statistical tests of cryptographic randomness. The internal state is initialized with a random value called a seed. The seed must have a high level of entropy to ensure that the stream of bits is sufficiently hard to guess.

Typically methods of gathering entropy include using seeding information gathered 410 from the local system 405 to seed and stir the PRNG 425. However, unless the seeding information gathered 410 from the local system 405 has sufficient entropy, an attacker can guess the output of the PRNG 415 with relative ease, and break into the system.

[0031] To provide further security, according to one embodiment of the present invention, additional seeding information is obtained 420 from one or more remote entropy servers 430 and 445, using a secured link 460. The remote entropy servers 430 and 445, which comprise random state machines 435 and 450, generate the additional seeding information 440 and 455. The process of securely obtaining seeding information 420 from one or more remote entropy servers 430 and 445 is repeated for redundant entropy servers. The additional seeding information generated 440 and 455, by the remote entropy servers 430 and 445, is gathered 420, in addition to the local seeding information 410, for the stirring process 425.

[0032] The stirring process 425 involves receiving and mixing of the gathered local seeding information 410, and remote seeding information 420. Using the combination of local and remote seeding information provides the unpredictable state that a system must have in order to fully secure the information. The security of a system depends on having a cryptographically secure PRNG algorithm. It is easy for an attacker to predict the state of a PRNG if only the local seeding information is utilized. However, with the stirring process 425 using local and remote seeding information 425, the much-needed entropy is amplified, making the system extremely secure, and difficult to break

into for the attacker. Thus, the stirring process 425 of the present invention provides security against cryptographic breaks when two applications communicate with each other, or even when information is sent from one computer to another over the Internet.

[0033] According to one embodiment of the present invention, secure data collection from entropy servers 420 is done using a privacy protocol, such as a Secure Sockets Layer (SSL) or Transport Layer Security (TLS). This prevents an attacker from getting a copy of the data supplied by the entropy server and reproducing the PRNG state on his machine. If the exchange is not done securely, its value could be greatly diminished. Additionally, privacy protocols, such as SSL and TLS, themselves require unpredictable random numbers to be secured. Thus, in environments requiring remote entropy servers, the privacy protocols may not be acceptable for securing the exchange, and therefore, an alternative may be required.

[0034] According to one embodiment of the present invention, an entropy server, which is a machine or piece of software, maintains a constantly updated random state pool that is used to supply hosts with seeding information that can be stirred into their PRNG state value. An attacker is more likely to be able to negatively influence the initial state seeding, and succeed when only one entropy server is used. Hence, according to one embodiment of the present invention, a local host may use more than one entropy server so that the attacker cannot influence the initial state seed by compromising a single entropy server.

[0035] **Figure 5** is a flow diagram illustrating a process for enhancing entropy

using a set of redundant entropy servers, according to one embodiment of the present invention. First, a PRNG is initialized in processing block 505. When a local host requires a PRNG, it seeds the initial state using locally unpredictable information in processing block 510. The local system seeding information is obtained in processing block 510. Further, seeding information is also securely obtained from one or more remote entropy servers in processing block 515. If there are no redundant servers in decision block 520, the PRNG is stirred in processing block 525. However, the process of obtaining seeding information is repeated for each redundant entropy server in decision block 520. According to one embodiment of the present invention, a local host may use more than one entropy server so that the attacker cannot influence the initial state seed by compromising a single entropy server. Finally, the PRNG is stirred using both the local and remote seeding information in processing block 525.

[0036] **Figure 6** is a block diagram illustrating logic for implementing a secure entropy collection protocol, according to one embodiment of the present invention. In situations in which it is undesirable to use a standard privacy protocol or when a standard privacy protocol is unavailable, a secure entropy collection protocol may be used to interact with one or more entropy servers. For instance, in an environment requiring remote entropy server, the privacy protocols may not be acceptable for securing the exchange, because the privacy protocols themselves require unpredictable random numbers to be secured.

[0037] According to one embodiment of the present invention, on the host-side

600, a temporary asymmetric key pair is generated 605. The temporary public key created 605 on the host-side 600 is then encrypted with a remote entropy server's public key 610. The encrypted public key is then sent to the remote entropy server 650. As discussed above, in a public-key system there is a corresponding private key to a public key. Generally, the private key is used to decrypt the corresponding public key's encrypted information. Thus, on the server-side 650, the host's temporary public key is then decrypted using the server's private key 620.

[0038] The server then generates random data 625, and encrypts it using the host's temporary public key 630. The encrypted random data is sent to the host 635. The random data is received on the host-side 600, and then decrypted using the host's temporary private key 640. Finally, the result of the decryption of the random data is used to stir the internal state of the local PRNG 645.

[0039] According to one embodiment of the present invention, random states from one or more external sources (e.g., redundant entropy servers) are added when gathering seeding information. This method provides additional security, because an attacker who is attempting to perform a cryptographic attack is likely to fail in predicting the random states from multiple external sources. In other words, the method allows the state of multiple independent systems to securely contribute to the strength of the local PRNG output. Thus, eliminating cryptographic breaks into a system by having strong and remote sources of randomness.

[0040] **Figure 7** is a flow diagram illustrating the process for implementing a

secure entropy collection protocol, according to one embodiment of the present invention.

In situations in which it is undesirable to use a standard privacy protocol or when a standard privacy protocol is unavailable, a secure entropy collection protocol may be used to interact with one or more entropy servers. First, on the host-side 600, a temporary asymmetric key pair is generated in processing block 705. The temporary public key created on the host-side is then encrypted with a remote server's public key in processing block 710. The encrypted public key is sent to the remote server in processing block 715. Then, on the server-side, the host's temporary public key is decrypted using the server's private key in processing block 720.

[0041] The server then generates random data in processing block 725, and encrypts it using the host's temporary public key in processing block 730. The encrypted random data is then sent to the host in processing block 735. The random data is received by the host, and then decrypted using the host's temporary private key in processing block 740. Finally, the result of the decryption of the random data is used to stir the internal state of the local PRNG in processing block 745.